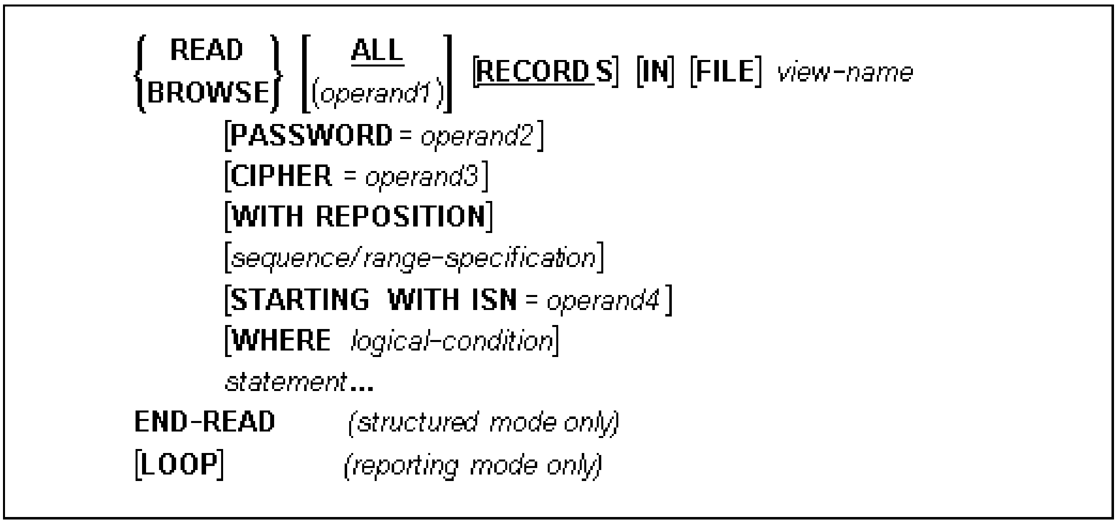
READ



```
{ READ   } [ ALL        ] [RECORDS] [IN] [FILE] view-name
{ BROWSE } [ (operand1) ]
        [PASSWORD = operand2]
        [CIPHER = operand3]
        [WITH REPOSITION]
        [sequence/range-specification]
        [STARTING WITH ISN = operand4 ]
        [WHERE logical-condition]
        statement...
END-READ        (structured mode only)
[LOOP]          (reporting mode only)
```

Operand	Possible Structure					Possible Formats												Referencing Permitted	Dynamic Definition
Operand1	C	S					N	P	I		B							yes	no
Operand2	C	S				A												yes	no
Operand3	C	S					N											yes	no
Operand4	C	S					N	P	I		B							yes	no

Related Statement: FIND | HISTOGRAM

Function

The READ statement is used to read records from a database. The records can be retrieved in physical sequence, in Adabas ISN sequence, or in the value sequence of a descriptor (key) field.

This statement causes a processing loop to be initiated.

Number of Records - operand1/ALL

The number of records to be read may be limited by specifying *operand1* (enclosed in parentheses, immediately after the keyword READ) - either as a numeric constant (0 - 99999999) or as a variable, enclosed within parentheses, immediately after the keyword READ. For example:

```
   READ (5) IN EMPLOYEES ...

   MOVE 10 TO CNT(N2)
   READ (CNT) EMPLOYEES  ...
```

For this statement, the specified limit has priority over a limit set with a LIMIT statement.

If a smaller limit is set with the LT parameter, the LT limit applies.

To emphasize that *all* records are to be read, you can optionally specify the keyword ALL.

Notes:
If you wish to read a 4-digit number of records, specify it with a leading zero: (0nnnn); because Natural interprets every 4-digit number enclosed in parentheses as a line-number reference to a statement.
Operand1 is evaluated when the READ loop is entered. If the value of operand1 is modified within the READ loop, this does not affect the number of records read.

view-name

As *view-name*, you specify the name of a view, which must have been defined either within a DEFINE DATA statement or outside the program in a global or local data area.

In reporting mode, *view-name* may also be the name of a DDM.

PASSWORD and CIPHER Clauses

These clauses are applicable only to Adabas or VSAM databases. They cannot be used with Entire System Server.

The PASSWORD clause is used to provide a password when retrieving data from a file which is password-protected.

The CIPHER clause is used to provide a cipher key when retrieving data from a file which is enciphered.

See the statements FIND and PASSW for further information.

WITH REPOSITION

This option can only be applied to VSAM databases.

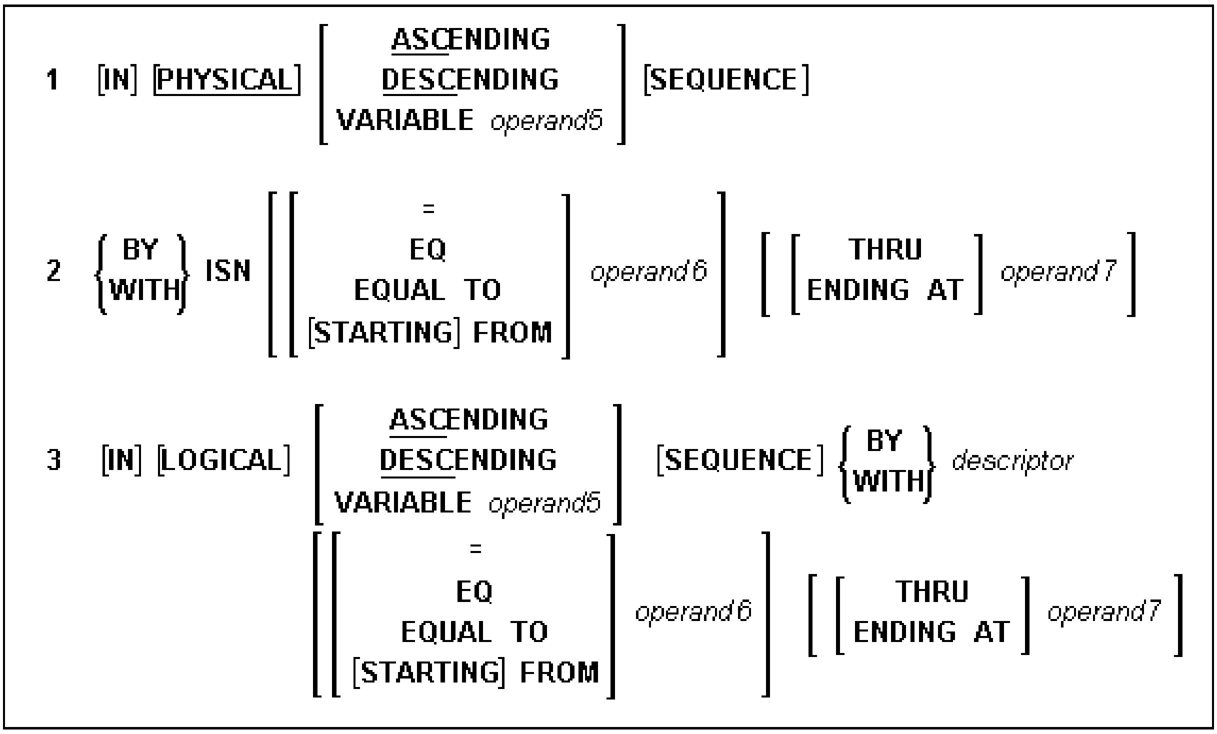
With this option, you can reposition to another start value for the database records read within the active READ loop. Processing of the READ statement then continues with the new start value.

The repositioning is triggered by the value of the system variable *COUNTER being reset to "0"; that is, the new start value is used as soon as *COUNTER is "0".

Example:

```
DEFINE DATA LOCAL
1 MYVIEW VIEW OF ...
  2 NAME
1 #STARTVAL (A20) INIT <'A'>
1 #ATTR     (C)
END-DEFINE
...
SET KEY PF3
...
READ MYVIEW WITH REPOSITION BY NAME = #STARTVAL
  INPUT (IP=OFF AD=O) 'NAME:' NAME /
    'Enter new start value for repositioning:' #STARTVAL (AD=MT CV=#ATTR) /
    'Press PF3 to stop'
  IF *PF-KEY = 'PF3'
    THEN STOP
  END-IF
  IF #ATTR MODIFIED
    THEN RESET *COUNTER
  END-IF
END-READ
...
```

sequence/range-specification



Options [2] and [3] are not available with Entire System Server.

Operand	Possible Structure					Possible Formats												Referencing Permitted	Dynamic Definition
Operand5		S				A												yes	no
Operand6	C	S				A	N	P	I	F	B	D	T	L				yes	no
Operand7	C	S				A	N	P	I	F	B	D	T	L				yes	no

READ IN PHYSICAL SEQUENCE

PHYSICAL SEQUENCE is used to read records in the order in which they are physically stored in a database.

For VSAM databases, READ PHYSICAL can only be applied to ESDS and RRDS.

PHYSICAL is the default sequence.

READ BY ISN

READ BY ISN can only be used for Adabas and VSAM databases; for VSAM databases, it is only valid for ESDS and RRDS.

READ BY ISN is used to read records in the order of Adabas ISNs (internal sequence numbers) or VSAM RBAs (relative byte addresses of ESDS) or RRNs (relative record numbers of RRDS) respectively.

READ IN LOGICAL SEQUENCE

LOGICAL SEQUENCE is used to read records in the order of the values of a descriptor (key) field.

If you specify a descriptor, the records will be read in the value sequence of the descriptor. A descriptor, subdescriptor, superdescriptor or hyperdescriptor may be used for sequence control. A phonetic descriptor, a descriptor within a periodic group, or a superdescriptor which contains a periodic-group field cannot be used.

If you do not specify a descriptor, the default descriptor as specified in the DDM (field "Default Sequence") will be used.

For **DL/I databases**, the descriptor used must be either the sequence field of a root segment, or a secondary index field. If a secondary index field is specified, it must also be specified in the PROCSEQ parameter of a PCB. Natural uses this PCB and the corresponding hierarchical structure to process the database.
As HDAM databases use a randomizing routine to locate root segments, no sensible results will be returned when using READ LOGICAL for HDAM databases; therefore you should use READ PHYSICAL for HDAM databases.

For **VSAM databases**, LOGICAL is only valid for KSDS with primary and alternate keys defined and for ESDS with alternate keys defined.

If the descriptor used for sequence control is defined with null-value suppression (Adabas only), any record which contains a null value for the descriptor will not be read.

If the descriptor is a multiple-value field (Adabas only), the same record will be read multiple times depending on the number of values present.

ASCENDING/DESCENDING/VARIABLE SEQUENCE

This clause only applies to Adabas, VSAM and SQL databases. In a READ PHYSICAL statement, it can only be applied to VSAM databases.

With this clause, you can determine whether the records are to be read in ascending sequence or in descending sequence.

- The default sequence is ascending (which may, but need not, be explicity specified by using the keyword ASCENDING).
- If the records are to be read in descending sequence, you specify the keyword DESCENDING.
- if you will determine at runtime whether the records are to be read in ascending or descending sequence, you specify the keyword VARIABLE followed by a variable (*operand5*). The value of *operand5* at the start of the READ processing loop then determines the sequence. *Operand5* must have the format/length A1 and can contain the value "A" (for "ascending") or "D" (for "descending").

Note for Adabas databases:
Descending sequence requires the following Adabas versions (or above): Version 3.1 on UNIX and Windows, Version 3.2 on OpenVMS, and Version 6.1 on mainframe computers.

Note for SQL databases:
On mainframe computers, the VARIABLE option cannot be used for SQL databases.

Example of DESCENDING Option:

```
READ EMPLOYEES IN DESCENDING SEQUENCE BY NAME = 'SMITH'
```

Example of VARIABLE Option:

```
DEFINE DATA LOCAL
1 #DIRECTION (A1) INIT <'A'>   /* 'A' = ASCENDING
1 #EMPVIEW VIEW OF EMPLOYEES
   2 NAME
   ...
END-DEFINE
...
IF *PF-KEY = 'PF7'
  THEN MOVE 'D' TO #DIRECTION
END-IF
READ #EMPVIEW IN VARIABLE #DIRECTION SEQUENCE BY NAME = 'SMITH'
  ...
END-READ
...
```

Further Examples:

See the programs READSCND and REAVSEQ in the library SYSEXRM.

STARTING FROM / ENDING AT

The STARTING FROM and ENDING AT clauses are used to limit reading to a set of records based on a user-specified range of values. The terms THRU and ENDING AT both imply an inclusive range. If a starting value is specified, reading will begin with the value specified. If the starting value does not exist in the file, the next higher value will be used. If no higher value exists, the loop will not be entered.

The keyword "EQ" or "=" only establishes a starting value for the read operation. The ending value must be specified with the ENDING AT option.

If the sequence descriptor is an Adabas hyperdescriptor, the ENDING AT clause must not be specified.

A multiple-value field must not be used with the ENDING AT option.

Note:
Internally, to determine the end of the range to be read, Natural reads one value beyond the ENDING AT value. If you use the last record read for further processing, be aware that this last record is in fact not the last record within

the ENDING AT range, but the first record beyond that range (except if there is no further value after the ENDING AT value).

STARTING WITH ISN=operand4

This clause applies only to Adabas and VSAM databases.

Access to Adabas

This clause can be used in conjunction with a READ statement in physical or in logical (ascending/descending) sequence. The value supplied (operand4) represents an Adabas ISN (Internal Sequence Number) and is used to specify a definite record where to start the READ loop.

Logical Sequence

Even if documented with an equal character "=", the READ statement does not return only those records with exactly the start value in the corresponding descriptor field, but starts a logical browse in ascending or descending order, beginning with the start value supplied. If some records have the same contents in the descriptor field, they will be returned in an ISN-sorted sequence.

The STARTING WITH ISN clause is some kind of a "second level" selection criterion that applies only if the start value matches the descriptor value for the first record.

All records with a descriptor value that is the same as the start value and an ISN that is "less equal"("greater equal" for a descending READ) than the start ISN are ignored by Adabas. The first record returned in the READ loop is either

- the first record with descriptor = start value and an ISN "greater" ("less" for a descending READ) than the start ISN,
- or if such a record does not exist, the first record with a descriptor "greater" ("less" for a descending READ) than the start value.

Physical Sequence

The records are returned in the order in which they are physically stored. If a STARTING WITH ISN clause is specified, Adabas ignores all records until the record with the ISN that is the same as the start ISN is reached. The first record returned is the next record following the ISN=start ISN record.

Access to VSAM

This clause can only be used in physical sequence. The value supplied (operand4) represents a VSAM RBA (relative byte address of ESDS) or RRN (relative record number of RRDS), which is to be used as a start value for the read operation.

Examples

This clause may be used for repositioning within a READ loop whose processing has been interrupted, to easily determine the next record with which processing is to continue. This is particularly useful if the next record cannot be identified uniquely by any of its descriptor values. It can also be useful in a distributed client/server application where the reading of the records is performed by a server program while further processing of the records is performed by a client program, and the records are not processed all in one go, but in batches.

For an example, see the program REASISND in the library SYSEXRM.

WHERE Clause

With the WHERE clause, you can specify an additional selection criterion in the form of a logical-condition. This criterion is evaluated **after** a record has been read and **before** any further processing (including AT BREAK processing) is performed on the record.

For details on logical condition criteria, see the Natural Reference documentation.

If a LIMIT statement or a processing limit is specified in a READ statement containing a WHERE clause, records which are rejected as a result of the WHERE clause are not counted against the limit.

System Variables

The following Natural system variables are available with the READ statement:

- ***ISN** - Contains the Adabas ISN of the record currently being processed.
  For VSAM databases, *ISN contains either the RRN (for RRDS) or the RBA (for ESDS) of the current record.
  For DL/I and SQL databases and with Entire System Server, *ISN is not available.
- ***COUNTER** - Contains the number of times the processing loop has been entered.

The format/length of these system variables is P10. This format/length cannot be changed.

The usage of the system variables is illustrated in the example on the next page.

Example 1

```
/* EXAMPLE ’REAEX1S’: READ (STRUCTURED MODE)
/*************************************************
DEFINE DATA LOCAL
1 EMPLOY-VIEW VIEW OF EMPLOYEES
  2 PERSONNEL-ID
  2 NAME
1 VEHIC-VIEW VIEW OF VEHICLES
  2 PERSONNEL-ID
  2 MAKE
END-DEFINE
LIMIT 3
/**************************************************
WRITE ’READ IN PHYSICAL SEQUENCE’
READ EMPLOY-VIEW IN PHYSICAL SEQUENCE
 DISPLAY NOTITLE PERSONNEL-ID NAME *ISN *COUNTER
END-READ
/**************************************************
WRITE / ’READ IN ISN SEQUENCE’
READ EMPLOY-VIEW BY ISN
     STARTING FROM 1 ENDING AT 3
 DISPLAY PERSONNEL-ID NAME *ISN *COUNTER
END-READ
/**************************************************
WRITE / ’READ IN NAME SEQUENCE’
READ EMPLOY-VIEW BY NAME
 DISPLAY PERSONNEL-ID NAME *ISN *COUNTER
END-READ
/**************************************************
WRITE / ’READ IN NAME SEQUENCE STARTING FROM ’’M’’’
READ EMPLOY-VIEW BY NAME
     STARTING FROM ’M’
 DISPLAY PERSONNEL-ID NAME *ISN *COUNTER
END-READ
/**************************************************
END
```

Equivalent reporting-mode example: See the program REAEX1R in the library SYSEXRM.

```
PERSONNEL         NAME           ISN       CNT
     ID
  --------- -------------------- --------- ---------

  READ IN PHYSICAL SEQUENCE
  50005600  MORENO                       2         1
  50005500  BLOND                        3         2
  50005300  MAIZIERE                     4         3

  READ IN ISN SEQUENCE
  50005800  ADAM                         1         1
  50005600  MORENO                       2         2
  50005500  BLOND                        3         3

  READ IN NAME SEQUENCE
  60008339  ABELLAN                    479         1
  30000231  ACHIESON                   884         2
  50005800  ADAM                         1         3

  READ IN NAME SEQUENCE STARTING FROM ’M’
  30008125  MACDONALD                  929         1
  20028700  MACKARNESS                 780         2
  40000045  MADSEN                     509         3
```

Example 2 - Combining READ with FIND

The following program reads records from the EMPLOYEES file in logical sequential order based on the values of the descriptor NAME. A FIND statement is then issued to the VEHICLES file using the personnel number from the EMPLOYEES file as search criterion. The resulting report shows the name (read from the EMPLOYEES file) of each person read and the model of automobile (read from the VEHICLES file) owned by this person. Multiple lines with the same name are produced if the person owns more than one automobile.

```
/* EXAMPLE 'REAEX2': READ AND FIND
DEFINE DATA
   LOCAL
1 EMPLOY-VIEW VIEW OF EMPLOYEES
  2 PERSONNEL-ID
  2 FIRST-NAME
  2 NAME
  2 CITY
1 VEH-VIEW VIEW OF VEHICLES
  2 PERSONNEL-ID
  2 MAKE
END-DEFINE
LIMIT 10
RD. READ EMPLOY-VIEW BY NAME STARTING FROM 'JONES'
     SUSPEND IDENTICAL SUPPRESS
FD.    FIND VEH-VIEW WITH PERSONNEL-ID = PERSONNEL-ID (RD.)
       IF NO RECORDS FOUND
          ENTER
       END-NOREC
       DISPLAY NOTITLE (ES=OFF IS=ON ZP=ON AL=15)
               PERSONNEL-ID (RD.) FIRST-NAME (RD.)
               MAKE (FD.) (IS=OFF)
       END-FIND
     END-READ
END
```

```
 PERSONNEL      FIRST-NAME         MAKE
      ID
--------------- --------------- ---------------

20007500        VIRGINIA        CHRYSLER
20008400        MARSHA          CHRYSLER
                                CHRYSLER
20021100        ROBERT          GENERAL MOTORS
20000800        LILLY           FORD
                                MG
20001100        EDWARD          GENERAL MOTORS
20002000        MARTHA          GENERAL MOTORS
20003400        LAUREL          GENERAL MOTORS
30034045        KEVIN           DATSUN
30034233        GREGORY         FORD
11400319        MANFRED
```